## **COMMAND LINE BASICS**

### **Introduction**

Feeling scared of the command line? You’re not alone. We have this image of developers staring intently at a black screen with white or green text flashing across as they wildly enter incomprehensible commands to hack into the corporate mainframe (no doubt while guzzling soda and wiping neon orange Cheetos dust off their keyboard).

That black screen or window is the command line interface (CLI), where you’re able to enter commands that your computer will run for you. While there’s no need for you to reenact the scene above, working with the command line is a critical skill for you to learn as a developer. The command line is like our base of operations, from which we can launch other programs and interact with them. It has a syntax of its own to learn, but since you’ll be entering the same commands dozens of times, you’ll quickly pick up the commands you need most.

In this introductory lesson to the command line, you’ll learn how to navigate around your computer and how to manipulate files and directories (also known as folders) directly from the comfort of the command line. You’ll soon see that this isn’t as difficult as you may think. The commands you will learn in this lesson are very straightforward, so don’t be intimidated by the prospect of using the command line for the first time.

#### 

#### 

#### 

#### **Test Drive Your Terminal**

Open a terminal on your computer. - Linux: open the programs menu and search for “Terminal”. You can also open the terminal by pressing CTRL + ALT + T on your keyboard. - MacOS: Open your Applications > Utilities folder and find “Terminal”.

Before we do anything, take a look at the following text: $ whoami This is a terminal command because it begins with a $. The $ is saying “Hey! Enter what follows in your terminal.” This means that we must exclude the $ when entering any command. In the example above, we would only enter whoami in our terminal. This is a common indicator so make sure that you aren’t entering $ before a command. Now that you are aware of what $ does, take your terminal for a test run! Make sure your terminal is open, type the command mentioned above, and press enter on your keyboard.

It returns your username. Cool!

#### **Why learn this now?**

You will be making heavy use of the command line throughout this curriculum, and the upcoming installation project will require you to install many different software programs using the command line. Additionally, you will primarily be using Git within the command line (more on this later). As part of the bigger picture, you may well be using the command line on a daily basis in your career as a software developer, making it an indispensable skill in your toolset.

### 

### **Learning Outcomes**

By the end of this lesson, you should be able to do the following:

* Describe what the command line is.
* Open the command line on your computer.
* Use the command line to navigate directories and display directory contents.
* Use the command line to create a new directory and a new file.
* Use the command line to rename or destroy a directory and a file.
* Use the command line to open a file or folder in a program.

### **Assignment**

Note: Many of these resources assume you’re using a Mac or Linux environment. If you did our previous installation lesson, you should already have Linux installed in dual-boot or a virtual machine. Or, you might be using MacOS. If you don’t have MacOS, or any version of Linux installed, please let me know.

1. Before diving into the command line lesson, you’ll want to know how to create a file. You can do so with the touch command. Open your terminal and enter ls (the l is a lowercase L). ls will show you the files and folders in the current directory (or will show nothing if the current directory is empty). Create a file called test.txt by entering this in your terminal: touch test.txt. Now enter ls once again. You should see test.txt listed in the output. You can also create more than one file at once using the touch command. Enter touch index.html script.js style.css and press the enter. Then enter ls once more. You should see the files in the output. Here is a small way that the terminal reveals its power. How long would it have taken to create all three of those files with your mouse? Thanks, terminal.
2. Read through [chapter 1 of Conquering the Command Line](http://conqueringthecommandline.com/book/basics).

#### **Use the Command Line Like a Pro**

There’s something important that you need to know about programmers. Programmers are lazy. Like, really lazy. If they are forced to do something over and over again, odds are good that they’ll figure out a way to automate it instead. The good news is that you get to benefit from the many shortcuts they’ve created along the way. It’s time to learn how to use the command line like a pro (which is to say, in a really lazy way).

First, you might have already noticed that copying and pasting inside the command line doesn’t work the way that you’d expect. When you’re inside the command line, you’ll need to use Ctrl+Shift+C (Mac: Cmd+C) to copy and Ctrl+Shift+V (Mac: Cmd+V) to paste. For example, to copy and paste commands from your browser into the command line, you’ll highlight the command text and use Ctrl+C as usual and then paste it in your terminal using Ctrl+Shift+V. Test it out!

Second, you need to learn about [tab completion](https://en.wikipedia.org/wiki/Command-line_completion). Seriously, this tip will save you so much time and frustration. Let’s say that you’re in the command line and that you need to move into a folder that’s far away, something like ~/Documents/Odin-Project/foundations/javascript/calculator/. That’s a long command to type out, and everything needs to be exactly right in order for it to work. Nope, we’re *way* too lazy for that! Basically, by hitting Tab, the command line will automatically complete commands that you’ve started typing once there’s only one option. For example, it’s pretty common to have a Documents folder and a Downloads folder in the home directory. If you’ve typed cd D and then press Tab, the command line will let you know that it’s not sure which one you want by showing you the different options that match what you’ve typed so far: bash $ cd D Documents/ Downloads/ $ cd D But once you’ve typed in a little bit more, it will complete the name for you, making it possible to write out the full file path above by typing as little as cd Doc[tab]O[tab]f[tab]j[tab]cal[tab] (depending on what other folders exist on your computer). Test it out, and get comfortable with how this works. You’re gonna love it.

Third, there’s a really handy shortcut for opening everything within a project directory: . Once you’ve installed a text editor, you can use this shortcut to open up an entire project and all of its files in one go. This shortcut is also commonly used with Git (which is covered in detail later on) with commands like git add . to add all of the files inside of a directory into Git’s staging area. For example, if you have VS Code installed, you can cd into the project directory and then type code . (with the period) to open up all of the project files. See the next section of this lesson for a more detailed example.

#### **Opening files in VSCode from the Command Line**

On Windows and Linux, you can open VSCode from the command line by typing code, and you can open folders or files by adding the name of the location after it: code my\_awesome\_project/.

##### MacOS Users:

MacOS can do this too, but you need to set it up. After installing VSCode, launch it any way you’re comfortable with. Once it’s running, open the Command palette with CMD + Shift + P. In the little dialog that appears, type shell command. One of the choices that appears will be Shell Command: Install 'code' command in PATH. Select that option, and restart the terminal if you have it open.

### **Exercise**

In this exercise, you will practice creating files and directories and deleting them. You’ll need to enter the commands for this exercise in your terminal. If you can’t recall how to open a terminal, scroll up for a reminder.

1. Create a new directory in your home directory with the name test.
2. Navigate to the test directory.
3. Create a new file called test.txt. *Hint: use the touch or echo command.*
4. Open your newly created file in VSCode and make some changes, save the file, and close it.
5. Navigate back out of the test directory.
6. Delete the test directory.

That’s it–you’re done with command line basics! If you commit to doing most things from the command line from here on out, these commands will become second nature to you. Moving and copying files is much more efficiently done through the command line, even if it feels like more of a hassle at this point.

### **Additional Resources**

This section contains helpful links to other content. It isn’t required, so consider it supplemental material if you want to dive deeper into something.

* The online book [Learn Enough Command Line to Be Dangerous](https://www.learnenough.com/command-line-tutorial) is a great resource for mastering the command line. Chapter 1 is free and provides a good introduction to command line tools. The rest of the book is not free and goes into more depth than you really need at this point, but feel free to purchase and read the rest of the book if you like.
* [ExplainShell.com](http://explainshell.com/) is a great resource for if you want to deconstruct a particularly strange shell command or learn how Bash works through guess-and-check.
* [Unix/Linux Command Cheat Sheet](https://files.fosswire.com/2007/08/fwunixref.pdf) contains a list of important commands that you can refer to regularly as you become familiar with using Linux. You can print it out so you can have a physical copy with you when you’re not at your computer.
* [Command Line Flashcards](https://flashcards.github.io/command_line/introduction.html) by flashcards.github.io.
* [The Video Series from LearnLinuxTv](https://www.youtube.com/playlist?list=PLT98CRl2KxKHaKA9-4_I38sLzK134p4GJ) contains 24 videos explaining the basics of the command line. Videos are brief enough for beginners but, at the same time, are detailed enough to get started and light your inner curiosity.

### **Knowledge Check**

This section contains questions for you to check your understanding of this lesson. If you’re having trouble answering the questions below on your own, review the material above to find the answer.

* What is the command line?
* How do you open the command line on your computer?
* How can you navigate to a particular directory?
* Where will cd on its own navigate you to?
* Where will cd .. navigate you to?
* How do you display the name of the directory you are currently in?
* How do you display the contents of the directory you are currently in?
* How do you create a new directory?
* How do you create a new file?
* How do you destroy a directory or file?
* How do you rename a directory or file?